ARL-TR-7710 ● JUNE 2016

US Army Research Laboratory

Network Science Research Laboratory (NSRL) Telemetry Warehouse

by Theron Trout and Andrew J Toth

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-7710 ● JUNE 2016

US Army Research Laboratory

Network Science Research Laboratory (NSRL) Telemetry Warehouse

by Andrew J Toth
Computational and Information Sciences Directorate, ARL

and

Theron Trout
Stormfish Scientific Corporation, Chevy Chase, MD

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
June 2016	Final	10/2014–09/2015

4. TITLE AND SUBTITLE
Network Science Research Laboratory (NSRL) Telemetry Warehouse

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Theron Trout and Andrew J Toth

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
US Army Research Laboratory
ATTN: RDRL-CIN-T
2800 Powder Mill Road
Adelphi, MD 20783-1138

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-7710

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Development of an architectural framework to validate performance of a distributed trust management protocol, called trustd, required a high-performance approach for experimentation data collection, storage, and retrieval. The Network Science Research Laboratory (NSRL) Telemetry Warehouse was developed by NSRL researchers to fulfill the needs of the trustd validation. This report describes the motivations and objectives of this project. Functionality and architecture of the NSRL Telemetry Warehouse are also described as well as the web interface, data structure, security aspects, and example deployments.

15. SUBJECT TERMS
Experimentation Database, data collection, data storage, data retrieval, trustd, US Army Research Laboratory, ARL, Network Science Research Laboratory, NSRL

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Andrew J Toth
Unclassified	Unclassified	Unclassified	UU	24	19b. TELEPHONE NUMBER (Include area code) 301-394-2746

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

INTENTIONALLY LEFT BLANK.

1. Overview

The US Army Research Laboratory (ARL) Network Science Research Laboratory (NSRL) is composed of a suite of hardware and software that models the operation of mobile networked device radio frequency (RF) links through emulation (not merely simulation). NSRL enables experimental validation or falsification of theoretical models, and characterization of protocols and algorithms for mobile wireless networks. It is used for a range of experiments, from assessing in-network aggregation of network information for detecting cyber threats, to characterizing the impact of communications disruption on perceived trust and quality of information metrics delivered to Soldiers in tactical mobile environments. Unlike other experimentation facilities for research in wireless networks, NSRL is focused on Army-unique requirements like hybrid networks and extensive modeling of ground and urban effects on communications. NSRL supports investigation of traditional wireless networking challenges as well as more general network science research issues. NSRL's emulation environment is result of collaborative efforts between ARL and the US Naval Research Laboratory (NRL).

The primary emulation tools used by ARL are the Extendable Mobile Ad-Hoc Network Emulator (EMANE)[1] and the Common Open Research Emulator (CORE).[2] Researchers at NSRL developed the NSRL Telemetry Warehouse (NTW) to improve the process of network science experimentation by providing a high-performance data collection and storage mechanism.

The software and systems running on the emulated networks execute in real time unlike simulations, which typically execute faster than real time by jumping from event to event skipping the time between events. One of the biggest advantages of emulated environments is that the characteristics of the network and its behaviors can be fully controlled and are repeatable. This is particularly interesting when emulating wireless networks as reproducibility of experimental environments using real radios is often difficult as temperature and humidity changes, differences in seasonal foliage, and other factors can alter the performance of the wireless networks. Figure 1 depicts the NTW within NSRL.

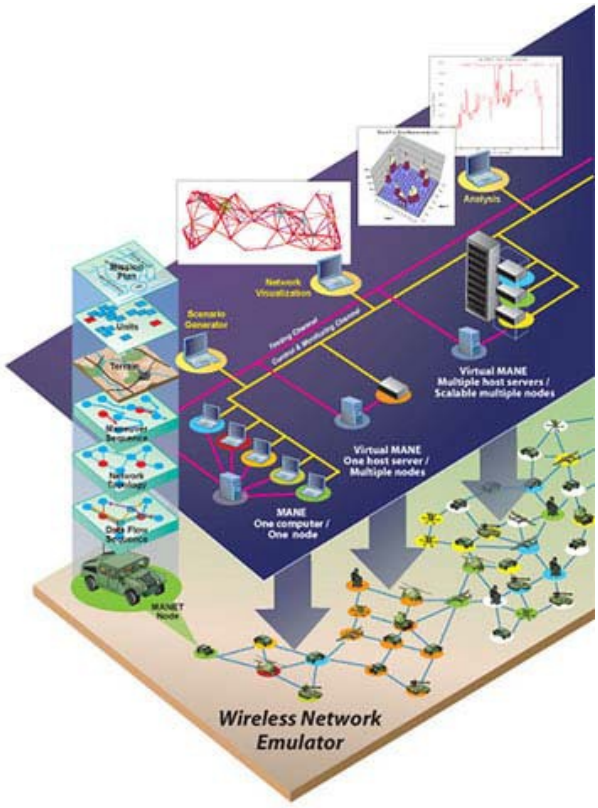

Fig. 1 NTW within NSRL

Simulated systems do not typically require high-speed data collection. While this is a desirable trait, the only impact of slower data collection is the overall duration of the simulation rather than results of the simulation. Emulations running in real time are sensitive to data collection timing constraints due to the criticality of event timing. Poor data collection performance negatively impacts emulation results because either data points will not be recorded or event execution will be delayed. The NTW provides a central location for collection, storage, and retrieval of experiment data in a generic, loosely structured form that meets the needs of real-time emulation.

2. Motivations and Objectives

The NTW is an outgrowth of trustd, a trust daemon experimental testbed for network emulation, developed by researchers in the NSRL.[3] The trustd system produces a considerable amount of data, which are used to make trust determinations. Recording and evaluating this volume of data necessitated a dependable data collection and storage system. The first attempt at building such a system was successful; however, it suffered from some stability and reliability issues that negatively impacted its usefulness.

The NTW leverages the Google remote procedure call (gRPC) system. The gRPC system uses the Google protocol buffers compiler to generate the code for the server. gRPC handles many of the basic elements of building server software systems, speeding the process of development. The result was that much of the code relevant to collecting, managing, and storing incoming telemetry could be moved from the previous system to the new gRPC-based solution. This made for a fast migration path. Most importantly, the gRPC system resolved nearly all of the stability and reliability issues experienced in the previous solution.

gRPC handles many of the nuts and bolts elements of building server software systems. This freed the developer to focus on features and capabilities. Most of this opportunity time was spent genericizing the NTW to remove any expectation of its use with the trustd system. The result is a system sufficiently generic as to be useful in a wide array of domains and applications.

3. NTW Functionality

3.1 Data Collection, Storage, and Retrieval

The NTW provides a central location for collection, storage, and retrieval of experiment data in a generic, loosely structured form. The NTW database provides facilities for storing up to 1 GB of configuration or initial condition information for each experiment to provide context for telemetry and support experiment reproducibility.

In order to run an experiment, the experiment controller must request the creation of an experiment session, which is linked to a current experiment run. Once the first experiment session has been created, the NTW server will deny any requests to change the definition or configuration data for the underlying experiment. If this were not the case, then subsequent runs of the experiment would not be comparable those performed previously. The user may simply duplicate the underlying experiment and make changes as desired when changes to the experiment are needed.

The “measurements” table stores the telemetry collected during an experiment run. This table is optimized for row insertion, not for data retrieval. Specifically, there is a minimum set of indexes defined for the columns in the table. Each time a record is inserted into a database table, the indexes must be updated (though this can be deferred to enhance performance, this is not a panacea). Updating the indexes takes time and slows the rate at which the database can insert new records.

3.2 Comparative Analysis

As mentioned previously, experiment definitions are not permitted to change once the first experiment run has been performed for that experiment. This is intended to provide a known set of initial conditions for each experiment and prevent changes that may preclude arriving at comparable results with subsequent experimental runs.

This characteristic of the NTW system enhances the ability for comparative analysis between runs to be performed. Users may retrieve the results of various runs of a certain experiment and compare them to determine which elements of the experiment are identical and which are different between runs.

Comparison between runs of different experiments can be performed, as well. This means that 2 experiments with differing initial conditions and configuration data may be prepared and run, and then their results compared.

3.3 Data Processing

Ultimately, the raw telemetry collected during the execution of an experiment must be converted to information useful to the researcher. The NTW provides access to the collected telemetry in comma-separated value (CSV) format from the web interface or via custom applications developed by researchers using the client application programming interface (API). Researchers can also query data directly from the database server.

4. Architecture

This section describes the NTW architecture. The architecture approach was driven by the need for high-speed data collection and storage in a distributed experimentation environment. The NTW includes the following:

- NTW database
- NTW server
- Experiment controller
- Sensors
- Custom data processing nodes
- NTW web interface

The relationships between these components are depicted in Fig. 2. Arrowheads identify the direction of data flow between the components.

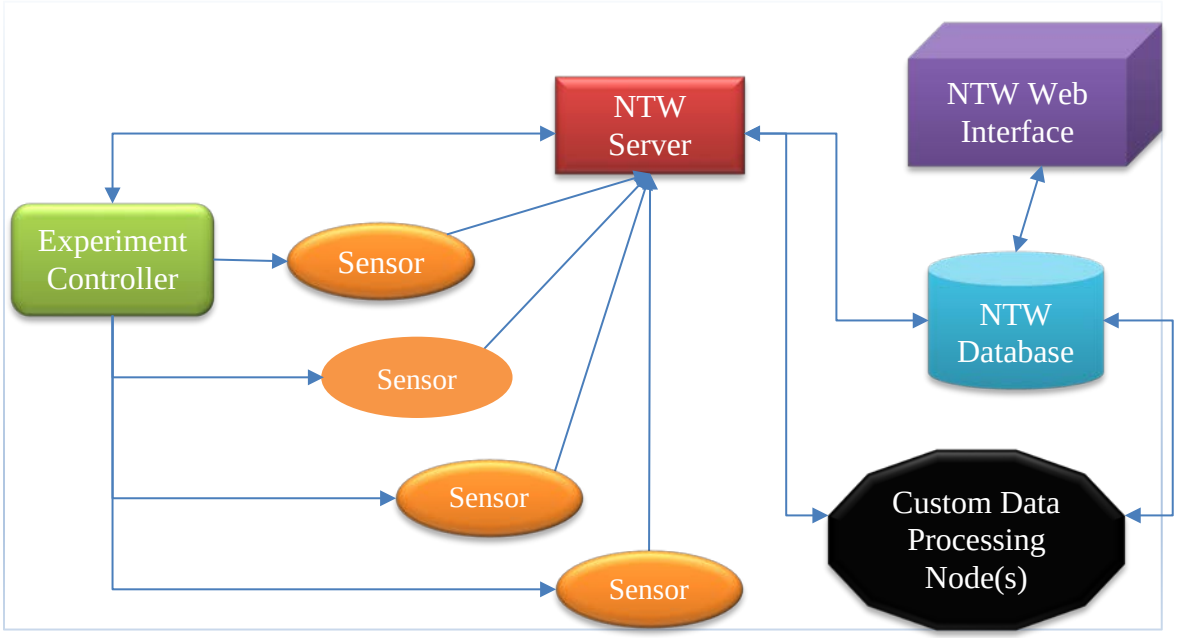

Fig. 2 NTW component architecture

Each component is described in detail below, but a brief description is appropriate here. The NTW database is the primary data store for all project and experiment details, as well as the storehouse for all collected telemetry. The NTW server provides the primary interface for users and systems to leverage the capabilities of the NTW. The server also enforces security and access restrictions for requests received.

Sensors are software elements that must be incorporated into the system under test. These are responsible for obtaining the telemetry of interest and sending it to the NTW server. The NTW client API provides tools to simplify the implementation of sensors.

The experiment controller is the primary interface through which experiment performers interact with the system. It will retrieve an experiment definition, if available, and configure experiment components in accordance with the definition. The controller will also distribute credentials to the sensors to allow them to send telemetry for the experiment session underway.

Custom data processing nodes may be employed to retrieve raw telemetry from the NTW system either as it is collected or as a postexperiment processing step. This is where raw telemetry can be converted to information more useful to the scientists

using the system. When operating during the experiment, these components can even be used to feed data back into the experiment, if appropriate.

A web interface for the NTW system provides a simple means of viewing data as it is stored in the warehouse. Data may be monitored in near real time using this interface, although it is not required for operation of the NTW. Access to the data is also available via structured query language (SQL) queries.

4.1 NTW Database

The NTW database is a PostgreSQL database server. PostgreSQL servers have wide-scale adoption, a long and trusted service record, and high performance and reliability features, which make it a good choice for the NTW system.

4.2 NTW Server

The NTW server is built atop the gRPC system. gRPC is an open-source product released by Google that is the latest incarnation of a similar system used in-house by Google to build many of its systems. Google is famous for providing efficient and responsive services to millions of simultaneous users. Seeing as their business model is largely dependent on maintaining its users' expected level of service and that gRPC is a primary component used to provide that service, it is a good choice for a telemetry collection system.

The NTW server manages user session, experiment runs, authenticates user credentials, and enforces access restrictions. It is, therefore, one of the most crucial components of the NTW architecture.

The NTW system uses a simplified keyed-hash message authentication code (HMAC) approach for ensuring data integrity and authenticity of messages. The current implementation of NTW uses HMAC-MD5 for simplicity in proof of concept; however, defaulting to more cryptographically secure hashing algorithms will be appropriate for production deployments. Preservation of HMAC-MD5 compatibility may remain desirable when running sensors of platforms with limited computational power.

4.4 Experiment Controller

The experiment controller is the primary interface to the NTW system used during the experimentation process. Generally, a custom experiment controller needs to be built to interface with the experimental systems under test.

The experiment controller can currently be built in 2 ways. The first option is to develop the controller in C++ and use the NTW client library to communicate with the NTW server. All details of authentication, session handling, and protocol messaging structures are handled for the developer.

The second method is to use the gRPC and protobuf message definition file to compile a custom client library in the user's preferred software development language. gRPC and protobuf support almost all of the most popular programming languages currently in use, including Java, Python, and C#.

Work is underway to provide Python bindings to the C++ NTW client library. This will provide a high-performance implementation of the client capabilities' need to interface with the NTW from Python.

4.5 Telemetry Sensors

In order to get telemetry data from an experiment into the NTW, the experiment designer must instrument their components to act as NTW sensors where desired. Fundamentally, this means sending protobuf messages containing measurement messages to the NTW server using the NTW client library.

NSRL researchers plan to develop Python and Java wrappers for this library.

Sensors must obtain an experiment session token in order to calculate a valid HMAC to write measurements to an experiment session. Use of HMAC ensure the integrity and authenticity of the telemetry messages received and recorded into the archive.

4.6 Custom Data Processing Nodes

Depending on the nature of the experiment, the telemetry stored in the NTW database will most likely be raw and unprocessed. Analyzing these data to transform them to information is usually required. This may be done manually, with CSV data downloaded through the NTW web interface, or automated by developing custom data processing nodes.

As with other components, data processing nodes can leverage client APIs, access the database directly, or construct their own gRPC and protobuf messages to send to the NTW server. Regardless of the data access methodology employed, these components are consumers of the telemetry stored in the NTW and use it to produce some meaningful output.

The NTW does not currently provide any facility specifically intended for storing processed results. Users will likely wish to save the processed information into

some data system of their own for incorporation into their work and analysis processes.

There are some interesting extended use cases for custom data processing nodes that might not be immediately obvious. For example, a processing node may calculate results from the telemetry as it arrives and then feed these resultant values back into the experiment. Alternatively, one might wish to feed previous results from another, related experiment into the current system under test. For example, perhaps a large, expensive exercise was performed and the telemetry captured. One may wish to feed telemetry from this exercise into the experiment as if it were occurring concurrently.

5. Web Interface

The web interface provides easy access to the NTW system. It uses D3.js to provide graphing capabilities of telemetry stored in the database. Telemetry data may be viewed in the web interface as soon as it is saved to the database, providing near-real-time review and monitoring of experiments.

As the NTW allows users to store nearly any data that they may collect by their instrumentation, the NTW system and D3.js are limited in what they can display. The current implementation of the web interface is currently limited to measurements with the data type name of “double” and where the value is a string containing a floating point value.

6. Data Structure

The NTW database stores telemetry measurements and contextual information about the experiment. The following sections highlight the most relevant data items stored in the database.

6.1 Measurements

Measurements are at the heart of the NTW and the reason for its existence. There are many ways in which one might categorize and tag telemetry data. The optimal structure for generic telemetry records remains an open question with work remaining in that domain. While it may not be a perfect fit for all types of telemetry that one might wish to store, the NTW data structure seems sufficiently flexible to handle a broad range of applications based on use in NSRL to date.

6.1.1 Platform

The “platform” is a text string that names a physical or logical entity making an observation. The platform name is required, is case sensitive, and may be up to 256 characters in length. Measurements that have matching platform names are considered to be from the same platform.

6.1.2 Sensor

The “sensor” is a text string that names a discrete sensor on a platform. The sensor name is required, is case sensitive, and may be up to 256 characters in length. Measurements with matching platform and sensor names are considered to be made by the same sensor.

6.1.3 Sensor Field

The “sensor field” is a text string that allows one sensor to record multiple values or fields. For example, a global positioning system sensor may output multiple fields including latitude, longitude, and altitude. Each of these would be a separate field. Like platform and sensor names, the field name is required, is case sensitive, and may be up to 256 characters in length. Measurements with matching platform, sensor, and field names are considered to be from the same sensor field.

6.1.4 Subject

The “subject” is an optional text string that identifies the entity that is the target of the measurement. For example, if the sensor were a radar speed gun, the subject might contain the license plate number of the vehicle whose speed is measured. The subject is *not* required, is case sensitive, and may be up to 256 characters in length. All matching subject strings in a particular experiment run are considered to identify the same subject entity in the experiment.

6.1.5 Data Type Name

The “date type name” is an optional character string that identifies the nature of the data stored in the value field of a measurement. If no data type name is given, the value “UNKNOWN” is recorded in the database. Common values for data type include, “integer”, “double”, etc. A convenient option is to store content type strings such as “application/json”, “application/xml”, “image/png”, and so forth. Alternatively, sensors may store any text string that is meaningful to them to identify the data structure stored in the “value” column.

“Data type name” is a case sensitive character string and may be up to 256 characters in length.

6.1.6 Value

The “value” field is used to store the measured telemetry value. It is required and must be encoded as text in a form appropriate for the value specified in “data type name”. The field can store up to 1 GB of data.

6.1.7 Extra Data

The “extra data” field is an optional storage space provided to store additional information about the measurement. This can be used for any purpose the user feels appropriate. Extra data must be encoded as text and may be up to 1 GB in length.

6.2 Data Organization

The NTW provides facilities to store a certain amount of information to give context to experiments. A primary benefit of this is the ability to organize and find experiments based on groups and projects in an organization.

6.2.1 Project

This represents a collection of related experiments. Projects include these data fields:

- Title
- Descriptions
- Project lead
- Extra data (optional)
- Date created

6.2.2 Experiment

An “experiment” in NTW specifies the definition for an experiment and includes the following data items:

- Experiment type
    - Project
- Title
- Description
- Extra data (optional)
- Owner user ID

    - The user who created the experiment
- Configuration (optional)
    - Up to 1 GB of storage for initial condition and any other details needed to define the experiment
    - Must be encoded as text

6.2.3 Experiment Session

An “experiment session” represents a “run” of an experiment. Telemetry values are tied to an experiment session, which, in turn, associates them with an experiment.

Once there is at least one experiment session associated with an experiment, changes to the configuration of the associated experiment are prohibited.

An “experiment session” includes the following data items:

- Owner user ID
    - Specifies who ran the experiment
- Experiment ID
    - Identifies the experiment which was run
- Title
    - A title describing the experiment run (e.g., “Network emulation run 25”)
    - Must be unique for each experiment
- Description
- Extra data (optional)
    - Storage space for additional information that the user may wish to record about the experiment run
    - Text format
    - 1 GB max
- Date created
    - Timestamp of when the experiment session was created

- Sensor authentication token
    - A random 128-bit integer
    - Sensors must have this token in order to record telemetry to this experiment session
- Date concluded
    - Timestamp identifying when the execution of the experiment was completed
    - No telemetry may be recorded to the experiment after this value has been set

7. Security

7.1 Access Control

The NTW system provides role-based access control. Users are assigned to security groups and permissions are assign to those security groups, which, in turn, are inherited by group members.

Multigroup membership grants permissions by applying logical OR permissions. If a user is a member of 2 groups, where one group has a particular permission and the other does not, the user will be granted the permission.

7.2 Experiment Session Tokens

Each experiment session is assigned a random, 128-bit integer session token. The NTW server will only record telemetry that includes an HMAC code generated with this token. This mechanism ensures the integrity and authenticity of telemetry recorded. If the sensor does not know the secret session token, it is unable to generate a valid HMAC to for the transmitted telemetry. Such records will be discarded by the server.

The current implementation of this feature is not complete. It was implemented to the degree that the concept could be validated. Additional steps are necessary to make this mechanism cryptographically secure. Specifically, the current implementation omits a mechanism to prevent replay attacks. There exist multiple ways of closing the replay-attack vulnerability; however, some care is needed in order to select a mitigation mechanism that does not restrict the simplicity of adding sensors to an experiment or computational overhead, which would preclude sensor deployments on low-power or central processing unit-limited systems.

8. Example Deployments

The trustd system is an architecture developed by NSRL for evaluating various algorithms and methodologies for trust-based routing. It has a pluggable architecture that allows a user to design and inject modules for different stages in network trust determinations.

The NTW has been effective in collecting data from various point in the trustd pipeline for post-run analysis and real-time monitoring. Additionally, the real-time retrieval capabilities of the NTW were effective in coupling the experiment runs to a run-time visualization of network-trust activity using the NRL Scripted Display Tool 3D.[4]

Other benefits of the NTW system is the simplicity that it has provided in comparing multiple runs of the same experiment while modifying particular elements of the experiment configuration. The NTW was effective in revealing an odd characteristic of the final trust values, which led to discovery of a significant bug in the code. Once this was corrected, it was easy to repeat the experiments using the new software version and compare the results.

9. Conclusion

The NTW has proved to be a benefit to network science experimentation in NSRL. Further use of this capability and sharing it with collaborators will increase our understanding of its utility and limitations. NSRL researchers will continue to develop the NTW and will make it available for public use on the ARL public website.[5]

10. References

1. US Naval Research Laboratory: extendable mobile ad-hoc network emulator (EMANE); 2016 [accessed 2016]. http://www.nrl.navy.mil/itd/ncs/products/emane.

2. US Naval Research Laboratory: common open research emulator (CORE); 2016 [accessed 2016]. http://www.nrl.navy.mil/itd/ncs/products/core.

3. Chan K, Cho JH, Chan K, Trout T, Wampler J, Toth A, Rivera B. trustd: trust daemon experimental testbed for network emulation. In: IEEE 2015. Proceedings of the Military Communications Conference (MILCOM); 2015 Oct 26–28; Tampa, FL. p. 641–646. doi: 10.1109/MILCOM.2015.7357516.

4. US Naval Research Laboratory: NRL scripted display tool 3D (SDT3D); 2016 [accessed 2016]. http://www.nrl.navy.mil/itd/ncs/products/sdt.

5. US Army Research Laboratory: NSRL homepage; 2014 Sep 15 [accessed 2016]. http://www.arl.army.mil/nsrl.

List of Symbols, Abbreviations, and Acronym

API	application programming interface
ARL	US Army Research Laboratory
CORE	Common Open Research Emulator
CSV	comma-separated value
EMANE	Extendable Mobile Ad-Hoc Network Emulator
gRPC	Google remote procedure call
HMAC	hash message authentication code
NRL	US Naval Research Laboratory
NSRL	Network Science Research Laboratory
NTW	NSRL Telemetry Warehouse
RF	radio frequency
SQL	structured query language

1 (PDF) DEFENSE TECHNICAL INFORMATION CTR DTIC OCA

2 (PDF) DIRECTOR US ARMY RESEARCH LAB RDRL CIO L IMAL HRA MAIL & RECORDS MGMT

1 (PDF) GOVT PRINTG OFC A MALHOTRA

1 (PDF) US ARMY RESEARCH LAB RDRL CIN A KOTT

3 (PDF) US ARMY RESEARCH LAB RDRL CIN T T TROUT A TOTH B RIVERA